Co-funded by the
Erasmus+ Programme
of the European Union

UNIDADE 3
EXPRESSÕES, PAUSAS E SONS

POLITEKNIKA IKASTEGIA TXORIERRI S.COOP.
isep Instituto Superior de Engenharia do Porto
TEI of Crete
Technological Educational Institute of Crete
apro

Objetivos e Conteúdos da Unidade 3

Objetivos

Analisar variáveis, constantes e expressões de em geral.
Introduzir funções novas da linguagem de programação Arduino para criar pausas e sons.

Conteúdos

- Apresentação do **tipo de expressões** que podem ser utilizadas com o Arduino
- As **funções básicas para utilização de pausas**
- As **funções básicas para utilização de sons**

Co-funded by the Erasmus+ Programme of the European Union

EXPRESSÕES

- Funções da linguagem Arduino que estudou até ao momento:
  - ✓ **pinMode(pin, mode)**: o número do pin que pretende tem de ser especificado e se é input ou output.
  - ✓ **digitalRead(pin)**: o número do pin que pretende que seja lido tem de ser especificado.
  - ✓ **digitalWrite(pin,value)**: o número do pin que pretende utilizer para estabelecer o valor (value) tem de ser especificado.
- Estes parâmetros podem ser especificados de formas diferentes:
  - ✓ **Constante**: digitalRead(4), lê o valor do input no pin número 4.
  - ✓ **Variável**: digitalRead(pushbutton), lê o pin de input definido na variável "pushbutton".
  - ✓ **Operação**:, digitalRead((1+1)*2) lê o pin 4 uma vez que este é o resultado de (1+1)*2.

Co-funded by the Erasmus+ Programme of the European Union

EXPRESSÕES

- CONSTANTES
  - Um **valor do programa que nunca é alterado** pode ser utilizado como uma "constante"
  - **NÃO pode ser alterado durante a execução** do programa
  - **Se decidir modificá-lo terá de alterar o programa** e gravá-lo novamente na memória do controlador
  - Estão **armazenadas na memória FLASH do controlador** do programa
  - **Não são voláteis** e mesmo quando a energia é desligada mantêm o seu valor (value)

EXEMPLOS

```
digitalRead(12);                 //lê o pin de input 12
digitalWrite(6,HIGH);            //estabelece o pin 6 no nível "1"
pinMode(9,OUTPUT);               //configura o pin 9 como output
```

Co-funded by the Erasmus+ Programme of the European Union

EXPRESSÕES

- VARIÁVEIS
  - Espaços entre a memória RAM do controlador especialmente desenhados para armazenar diferentes tipos de dados
  - Pode ler-se e escrever-se na memória RAM o número de vezes que se quiser, desde que de forma correta

EXEMPLO

type name = value

type: *Estabelece o tipo de dados que a variável contém*
name: *Este é o nome atribuído à variável ou recetáculo*
value: *Este é o conteúdo ou valor incluído na variável*

Co-funded by the Erasmus+ Programme of the European Union

Resumo do tipo de variáveis mais comum

TIPO	BYTES	DESCRIÇÃO
char	1	Um char utiliza 1 byte de memória e armazena o valor de um carater (8 bits). Os carateres literais são escritos entre aspas simples (‘). O tipo de dados char é um tipo assinalado, isto é, que codfica números desde -128 até 127.
byte	1	Um byte armazena um número *unsigned* de 8 bits e produz uma média de 0 a 255 (28-1).
int	2	*Integers* são o tipo de dados prioritário para o número de armazenamento. Armazenam um valor atribuído de 16-bit (2-byte) e rentabilizam uma média de -32768 a +32767.
unsigned int ou word	2	Armazena um número de 16-bit não atribuídos e rentabiliza uma média entre 0 e 65535 (216-1)
long	4	Estas são variáveis atribuídas que armazenam 32 bits (4 bytes) e rentabilizam uma média de -2,147,483,648 a 2,147,483,647.
unsigned long	4	Variável não atribuída que armazena 32 bits (4 bytes) e rentabiliza uma média de 0 a 4,294,967,295 (2^32 - 1).
float	4	Os números de pontos flutuantes (floating-point) podem ser tão elevados como 3.4028235E+38 e tão baixos como -3.4028235E+38. São armazenados como 32 bits (4 bytes) de informação.

Co-funded by the Erasmus+ Programme of the European Union

EXPRESSÕES

- **“Arrays” são outro tipo de variável**
  - ✓ Um array é um grupo de variáveis e pode incluir qualquer um dos tipos considerados na tabela anterior.
  - ✓ Para aceder aos conteúdos das variáveis é necessário indicar o seu número indexado

- **As variáveis podem ser “globais” ou “locais” dependendo do local onde são declaradas no programa**
  - ✓ **Globais:** variáveis declaradas ou definidas for a de todas as funções no programa, incluindo setup() e loop()
  - ✓ **Locais:** variáveis declaradas e criadas dentro de uma determinada função e só podem ser utilizadas nessa função

Co-funded by the Erasmus+ Programme of the European Union

EXPRESSÕES

- OPERAÇÕES
  - Os valores inseridos nas variáveis podem ser obtidos como resultado de todo o tipo de operação aritmética entre variáveis e/ou constantes.

Estas são as operações matemáticas mais comuns

=	**Equivalência**
+	**Soma**
-	**Subtração**
*****	**Multiplicação**
/	**Divisão**
%	**Remanescente (remanescente de uma divisão entre dois números inteiros)**

Co-funded by the Erasmus+ Programme of the European Union

TEMPOS

- Normalmente é **boa idea que o controlador não faça absolutamente nada** de útil, ou, por outras palavras, que **desperdice um pouco do seu tempo.**
- A linguagem de programação Arduino tem um conjunto de funções que colocam em **pausa a execução** de um programa durante um determindao período de tempo

➢ FUNÇÃO DELAYMICROSECONDS()

- Coloca o programa em pausa pelo período de tempo especificado (em microssegundos)

delayMicroseconds(n);

n: *indica o número de milisegundos que pretente ter o programa em pausa. É um* unsigned integer *de 16 bit.*

EXAMPLE

```
A = 100;
delayMicroseconds(A);  //pausa o programa durante 100 µS = 0.1 mS = 0.0001 segundos
```

Co-funded by the Erasmus+ Programme of the European Union

EXPRESSÕES

- FUNÇÃO DELAY()
  - Pausa o programa durante os milissegundos indicados

delays(n);

*n: indica o número de milisegundos (mS) que pretende que o programa esteja em pausa. São referidos como números longos sem sinal de 32 bits ou números longos não atribuídos (*unsigned long numbers*)*

EXEMPLO

```
A = 100;
delay (A);      //Pausa o programa durante 100 mS = 0.1 segundos de delay(1000)
```

Co-funded by the Erasmus+ Programme of the European Union

EXPRESSÕES

- FUNÇÃO MICROS()
  - Returns the number of microseconds elapsed since Arduino began running your program
  - It returns an unsigned long value and the number will overflow (go back to zero) after approximately 70 minutes

var = micros();

var: Esta variável armazena os microssegundos (µS) decorridos desde que o sistema for a reiniciado.

Co-funded by the Erasmus+ Programme of the European Union

EXPRESSÕES

- FUNÇÃO MILLIS()
  - Retorna o número de milissegundos (mS) decorridos desde que o Arduino começou a correr o programa
  - Retorna um valor longo não atribuído e o número atinge o alotamento (volta a zero) depois de aproximadamente 50 dias

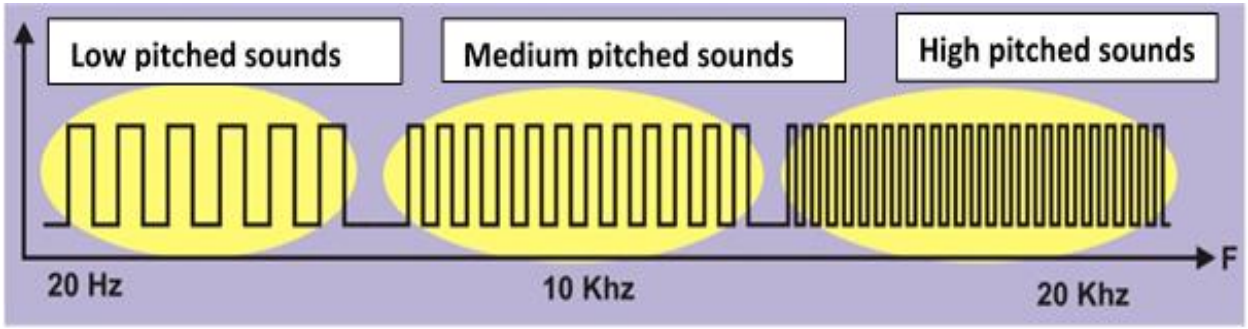
var = micros();

var: Esta é a variável que armazena os milissegundos (mS) decorridos desde o reiniício do sistema.

Co-funded by the Erasmus+ Programme of the European Union

O SOM

- O que precisa para criar som?
  - ✓ Um aparelho que possa vibrar e produzir alterações de pressão ou vibrações no ar
  - ✓ Estas vibrações atingem os nossos tímpanos e o nosso cérebro entende-os como som
  - ✓ Existem diversos aparelhos conhecidos que provocam alterações na pressão do ar e que por isso criam ondas que transportam o som: altifalantes, auscultadores, buzzers, sirenes
  - ✓ Os seres humanos conseguem ouvir o som das frequências entre aproximadamente os 20 Hz e os 20,000 Hz (20 KHz) ou ciclos por segundo. Isto indica a frequência ou número de vezes por segundo que um sinal passa de “1” a “0”.



Co-funded by the Erasmus+ Programme of the European Union

FUNÇÕES DO SOM

- A linguagem de programação Arduino facilita a produção de todos os tipos de som
  - ✓ Tudo o que há a fazer é **indicar a frquência F** e o **Arduino trata do resto!**

➢ FUNÇÃO TONE()

- Gera um som que requer frequência F no pin de output durante o período indicado.

tone(pin, frequency, duration);

pin: indica qual o pin onde se vai gerar o som.

frequency: representa a frequência do som em Hz (hertz).

duration: Este parâmetro é opcional. É um número longo unsigned que representa a duração do som em milissegundos.

Co-funded by the Erasmus+ Programme of the European Union

FUNÇÕES DO SOM

- FUNÇÃO noTONE()
  - Pára um som num determinado pin

tone(pin);

pin: representa em que pin o som vai ser parado.

EXEMPLO:

```
tone(2,13000);          //Gera um som indefinido de 13 KHz no pin 2.
....                     //O som mantem-se.

noTone(2);              //O som pára.
```

Co-funded by the Erasmus+ Programme of the European Union

Co-funded by the
Erasmus+ Programme
of the European Union

UNIDADE 3 EXPRESSÕES, PAUSAS E SONS

Obrigado!

POLITEKNIKA IKASTEGIA TXORIERRI S.COOP.
isep Instituto Superior de Engenharia do Porto
TEI of Crete
Technological Educational Institute of Crete
apro